LISTING OF THE CLAIMS:

1. (Currently Amended) A method, comprised of enhancing a computational service to each client of a plurality of clients, by:

moving a selected portion of a computation from a server into a trusted co-server executing inside a secure coprocessor;

allowing each client to interact with the server and the co-server; and

using the trusted co-server as a trusted third party ~~in~~ to authenticate interactions between the client and the server.

2. (Original) A method as in Claim 1, wherein the step of allowing includes providing a trusted third party at said server.

3. (Original) A method as recited in Claim 1, wherein said step of allowing includes enabling said client an authenticated, private channel to said co-server.

4. (Original) A method as in Claim 1, wherein said service is a Web service and said clients are remote users operating browsers.

5. (Original) A method as in Claim 3, wherein said step of enabling includes the client using the co-server's certified keypair to establish a shared symmetric key.

6. (Original) A method as in Claim 5, wherein said step of enabling includes employing the Secure Sockets Layer (SSL) protocol.

7. (Original) A method as in Claim 1, wherein said step of moving includes integrating functions of said co-server in a same machine as said server.

8. (Original) A method as in Claim 1, wherein said step of enhancing includes providing a desired security and/or privacy property.

9. (Original) A method as in Claim 1, wherein said step of enhancing includes providing at least one security and/or privacy property to an application selected from the group including: authentication of clients, nonrepudiation of client activity, nonrepudiation of server activity, credit card transaction security, taxes on e-commerce activity, re-selling of intellectual property, privacy of sensitive or proprietary web activity, correctness of web activity, enforcement of logo and/or "seal of approval" licenses, safety of downloadable content, authenticity of downloadable content, integrity of server machine, and any combination of these.

10. (Previously Presented) A method as in Claim 1, wherein:

input from said client is prompt from server for the user's private authenticator data, input from said server is this authentication data, co-server algorithm that generates output to a client based on said current co-server state and said inputs indicates whether or not the authenticator data is correct for this user.

11. (Previously Presented) A method as in Claim 1, where co-server algorithm that generates output to said server based on a current co-server state and inputs includes a signed statement, using a private key known to the co-server, attesting, for the server, that the client engaged in an interaction satisfying certain properties.

12. (Previously Presented) A method as in Claim 1, where co-server algorithm that generates output to said client based on a current co-server state and inputs includes a signed statement, using a privacy key known to the co-server, attesting, for the client, that the server engaged in an interaction satisfying certain properties.

13. (Previously Presented) A method as in Claim 1, wherein:

the client's input includes a credit card number (CCN), the output co-server algorithm that generates output to said client based on a current co-server state and inputs includes the CCN, encrypted so that the server cannot read it but an acquirer can.

14. (Previously Presented) A method as in Claim 13, wherein:

the server's input includes a transaction amount, the output co-server algorithm that generates output to said client based on a current co-server state and inputs includes the transaction amount, cryptographically bound to the encrypted CCN so that the server cannot alter it.

15. (Original) A method as in Claim 1, where:

the client's input includes a credit card number, the server's input includes a transaction amount, the co-server encrypts this CCN so that the server cannot read it but an acquirer can, and cryptographically binds the transaction amount to the this encrypted CCN, then, at some point during or after the interaction, transmits this data to the acquirer in such a manner so that the acquirer can receive this transaction exactly once.

16. (Previously Presented) A method as in Claim 1, wherein:

the interaction via the server input and/or the client input, includes a transaction amount A, the co-server input may include an accumulated total, the function co-server algorithm that generates new co-server state based on a current co-server state and inputs updates the accumulated amount by adding T(A), where T is a predefined function, and at some point during or after this interaction, the co-server produces an authenticated statement of the current value of the accumulated amount.

17. (Previously Presented) A method as in Claim 1, where:

a remote party is an owner of intellectual property, the server input includes part of this property, encrypted so that only the co-server can decrypt it, the output function co-server algorithm that generates output to said client based on a current co-server state and inputs to the client includes a portion of a decryption of input from said client.

18. (Original) A method as in Claim 17, except the output function co-server algorithm that generates output to said client based on said current co-server state and said inputs now includes a transformation of a portion of the decryption of input from said server, where said transformation may include adding a watermark.

19. (Original) A method as in Claim 17, except the output function now includes a transformation of a portion of the decryption of input from said server, where said transformation may include reducing the quality of the plaintext.

20. (Original) A method as in Claim 17, except the output function now includes a portion of the decryption of input from said server, re-encrypted, possibly with rights management rules, in a manner that a secure coprocessor at the client site can decrypt it.

21. (Original) A method as in Claim 1, wherein:

the client input includes a choice of which record R in a set of records the client would like to receive, the co-server includes this record R in its response to the client, however, the co-server obtains R in such a way as the server does not know which record was the one selected.

22. (Previously Presented) A method as in Claim 1, wherein:

a remote party establishes a content evaluation scheme, consisting of an evaluation function mapping content to some set of indicators, and as part of computing the client output

function co-server algorithm that generates output to said client based on a current co-server state and inputs, the co-server calculates, or verifies an external calculation, of the evaluation function and includes the result in the client output.

Claims 23 and 24 (Cancelled).

25. (Original) A method as in Claim 24, where the evaluation function is parameterized by a "signature file" and where the client output includes an identification of which signature file was used in this interaction.

26. (Original) A method as in Claim 22, where party the remote party has injected evaluation function and/or some of its parameters into the co-server through a private channel, so that the server cannot know the details of the evaluation function execution occurring on the co-server.

27. (Original) A method as in Claim 22, where the server input includes both content and a signature on the content, from one of possibly many content providers, and the evaluation function includes testing whether the signature is valid.

28. (Previously Presented) A method as in Claim 1, where:

a remote party establishes a content evaluation scheme, consisting of an evaluation function mapping content to some set of indicators, and as part of computing the server output function co-server algorithm that generates output to said client based on a current co-server state

and inputs or internal function co-server algorithm that generates new co-server state based on said current co-server state and said inputs the co-server calculates, or verifies an external calculation, of the evaluation function input from said client and includes the result in the output.

29. (Previously Presented) A method as in Claim 1, where:

the co-server has the ability to carry out security-enhancing actions against the server, and the output returned to client indicates which of these actions have been carried out, and how recently.

30. (Previously Presented) A method as in Claim 1, where:

the client can specify whether the interaction is a read interaction or a write interaction;

for a write interaction:

the client input includes a message M and a specification S of the appropriate entities who can read this message;

the co-server retains M and S by storing them in some combination across the co-server and server via an algorithm that generates new co-server state based on said current co-server state and said inputs, the internal state in the co-server and co-server algorithm that generates output to said server based on a current co-server state and inputs;

however in said write interaction:

any portion of M sent via co-server algorithm that generates output to said server based on said current co-server state and said inputs is encrypted, so that the server cannot access the plaintext;

and mechanisms are used to ensure that, when the co-server later retrieves any of this data from the server, that the data has not been changed;

for a read interaction:

the client input specifies which message M the client would like to read, the co-server retrieves S; if the client satisfies S, then the co-server sends M back to the client, after first retrieving and decrypting it, if necessary.

31. (Currently Amended) A method for enhancing a service to provide security and/or privacy to each client from a plurality of clients, said service including computation on a server controlled by an operator, the method comprising:

moving a selected portion of said computation from a server controlled by said operator into a trusted co-server executing inside a secure coprocessor;

allowing clients to interact with the server through the co-server; and

using the trusted co-server as a trusted third party ~~in~~ to authenticate interactions between the client and the server.

32. (Original) A method as recited in Claim 31 wherein the secure coprocessor is installed at the server.

33. (Currently Amended) A method for enhancing a service including computation on a server controlled by an operator, the method comprising:

providing at least one security and privacy property to at least one client from a plurality of clients by:

moving a selected portion of said computation from a server controlled by said operator into a trusted co-server executing inside a secure coprocessor;

enabling clients to interact with the server and the co-server; and

using the trusted co-server as a trusted third party ~~in~~ to authenticate interactions between the client and the server.

34. (Currently Amended) A trusted co-server, executing a program such that:

for multiple parties, including a Web server, a remote client and said co-server, each party ~~may, optionally, provide~~ provides input, and then the co-server carries out for each party, a function on all these inputs, and ~~optionally returns~~ output to said each party; and

wherein the co-server executes so as to authenticate interactions between the client and the Web server so that said parties can authenticate and trust the correct execution of the co-server, in interactions between the client and the ~~server~~ co-server, despite attempts by the Web server to subvert ~~this~~ said execution.

35. (Original) A trusted co-server according to Claim 34, wherein the co-server executes inside a tamper respondent secure coprocessor.

36. (Original) A trusted co-server according to Claim 34, wherein the secure coprocessor is co-located at said server.

37. (Currently Amended) A method of enhancing the security of a Web based transaction utilizing a server, the method comprising the steps:

providing the server with a trusted co-server; and

using the trusted co-server to execute a program such that:

for multiple parties,

each party ~~may, optionally, provide~~ provides input and then said co-server carries out for each party, a function on all these inputs to authenticate interactions between the party and the server and the parties trust interactions between the parties and the server.

38. (Original) A method according to claim 37, where:

one party is a Web server and another party is a remote client.

39. (Previously Presented) A method according to Claim 37, where:

the client authenticates the co-server, the client sends its input to the co-server over a private channel, the co-server sends its output to said another party over a private channel, such as one established by encryption with a shared secret key.

40. (Currently Amended) A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform method steps for enhancing a computational service to at least one client of a plurality of clients, said method steps comprising:

moving a selected portion of a computation from a server into a trusted co-server executing inside a secure coprocessor; ~~and~~

allowing each client to interact with the server and the co-server; and

using the trusted co-server as a trusted third party ~~in~~ to authenticate interactions between the client and the server.

41. (Original) A program storage device according to Claim 40, wherein the step of allowing includes providing a trusted third party at said server.

42. (Original) A program storage device according to Claim 41, wherein the step of allowing includes enabling said client an authenticated, private channel to said co-server.

43. (New) A method according to Claim 1, wherein:

the moving step includes the step of an operator of the server using a secure coprocessor platform to install and certify the trusted co-server, including the steps of

i) the server operator obtaining a secure coprocessor platform,

ii) the server operator installing co-server application software into the secure coprocessor platform, said co-server application software having an ability to authenticate itself using a key pair of the co-server,

iii) the co-server application then generating a new key pair including a public key and a private key,

iv) the server operator then using the co-server application's ability to authenticate itself to prove to a specified secure socket layer (SSL) certificate authority that said new key pair belongs to an installation of said co-server application running on an untampered secure coprocessor platform at the site of said server operator,

v) the SSL certificate authority then issuing an SSL-compatible certificate attesting to the public key of said generated new pair and the entity to which said public key belongs, and

vi) the co-server application storing said certificate; and

the using step includes the steps of

i) establishing an SSL session between the client and the trusted co-server,

ii) the client using a Web browser to initiate an SSL session with the co-server application within the secure co-processor at a Web site maintained by the server operator, and

iii) said Web browser indicating to the client that the co-server application demonstrates knowledge of the private key of said generated new key pair.

44. (New) A method according to Claim 43, wherein the using step includes the further steps of:

the client opening an SSL session to the trusted co-server, said trusted co-server being configured with a payment application, including the steps of

i) the server forwarding a price to the co-server,

ii) the co-server then displaying said price and accepting private credit card information of the client,

iii) the co-server signing and encrypting said price and said private credit card information, and

iv) the server operator then injecting said encrypted price and credit card information into a payment system;

the client opening an SSL session with a trusted co-server configured with a server status application, including the step of the co-server displaying authenticated information to the client about the server and providing a link by which the client can connect to the server; and

the client opening an SSL session with the trusted co-server, said trusted co-server being configured with an authentication application, including the steps of

i) the co-server prompting the client for client authentication information, including a used id and password,

ii) the client providing said authentication information,

iii) the co-server verifying the authenticity of said information, then directing the client to the server, and providing the server with an authentication token indicating that ht client has properly authenticated.